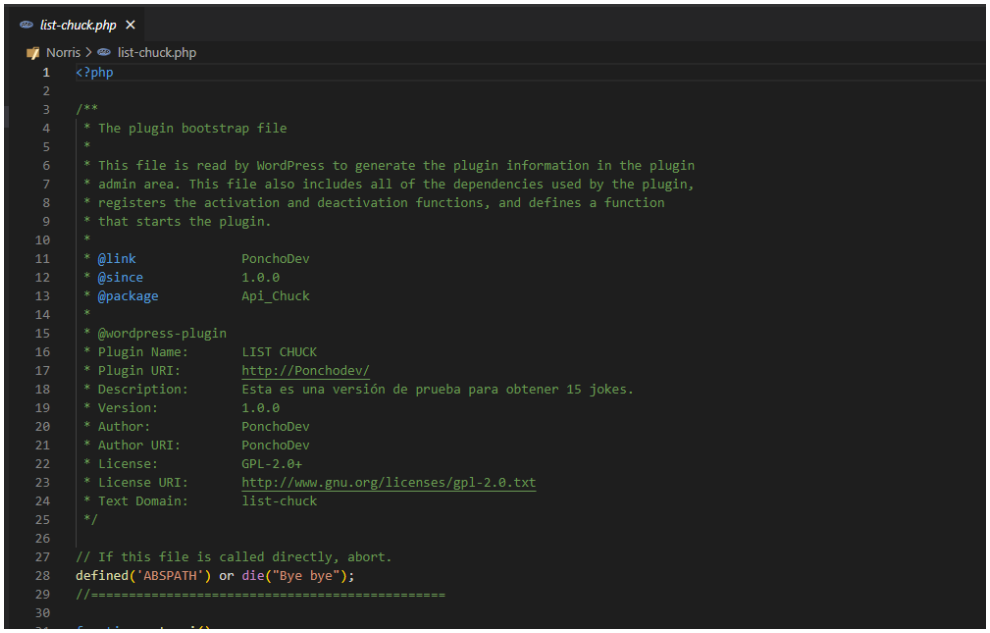
Creando un Plugin para WordPress el cual consume recursos en forma de texto desde un API

Iniciamos------- primeras líneas configuración del plugin:



```php
<?php

/**
 * The plugin bootstrap file
 *
 * This file is read by WordPress to generate the plugin information in the plugin
 * admin area. This file also includes all of the dependencies used by the plugin,
 * registers the activation and deactivation functions, and defines a function
 * that starts the plugin.
 *
 * @link              PonchoDev
 * @since             1.0.0
 * @package           Api_Chuck
 *
 * @wordpress-plugin
 * Plugin Name:       LIST CHUCK
 * Plugin URI:        http://Ponchodev/
 * Description:       Esta es una versión de prueba para obtener 15 jokes.
 * Version:           1.0.0
 * Author:            PonchoDev
 * Author URI:        PonchoDev
 * License:           GPL-2.0+
 * License URI:       http://www.gnu.org/licenses/gpl-2.0.txt
 * Text Domain:       list-chuck
 */

// If this file is called directly, abort.
defined('ABSPATH') or die("Bye bye");
//============================================
```

Aquí definimos el plugin, un plugin básico con descripción y nombre, algo importante para que WordPress pueda instalarlo correctamente.

```php
function get_api()
{

    $ch = curl_init();

    curl_setopt($ch, CURLOPT_URL, "https://api.chucknorris.io/jokes/random");
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
    curl_setopt($ch, CURLOPT_HEADER, false);

    curl_setopt($ch, CURLOPT_HTTPHEADER, array(
        "Content-Type: application/json"
    ));

    $response = curl_exec($ch);
    curl_close($ch);

    if (!$response) {
        exit('Error en API origen');
    }

    // Obtenemos datos de la API de origen
    $items_origin = json_decode($response, true);

    return $items_origin;
}
```

esta función hace Get al API y nos trae un json que decodificamos

```php
// Add Shortcode
function custom_chuck()
{

    ob_start();  //guarda en memoria todo la ejecución siguiente

    $count = 0;

    $arrSet = [];
    $arrResult = [];

    while ($count < 15) {
        $resulArrAPI = get_api();

        if (!in_array($resulArrAPI['id'], $arrSet)) {

            $arrResult[$count] = [
                'title' => $resulArrAPI['value'],
                'url' => $resulArrAPI['url'],
                'icon' => $resulArrAPI['icon_url']
            ];
            $arrSet[$count] = $resulArrAPI['id'];
            $count++;
        }
    }


    //echo "<p> <a href='".$arrResult[0]['url']."' target='_blank'>". $arrResult[0]['title']."</a></p>";

    echo "<ol class='footer'>";
    foreach ($arrResult as $singleVal) {
        echo "<li> <a href='" . $singleVal['url'] . "' target='_blank'>" . $singleVal['title'] . "</a></li>";
    }
    echo "<ol>";
?>
    <style type="text/css">
        .footer {
            margin-top: 40px;
        }
```

```php
?>
    <style type="text/css">
        .footer {
            margin-top: 40px;
        }

        .footer:before {
            content: '';
            position: relative;
            bottom: -8px;
            display: inline-block;
            background-image: url(
            height: 50px;
            width: 50px;
        }
    </style>

<?php


    $contentResult = ob_get_clean(); //Obtien el resultado que está en memoria

    return $contentResult; //Retornamos lo obtnido

}
```

En esta función procesamos el resultado de la API el cual recorremos con un while y pintamos los resultados en un “ol-list” de “Html”

```php
//Esta función crea un campo personalizado con el valor del shortcode
//Yse adiciona como valor a nuestra API de woocomerce
function save_field_custom($post_id)
{

    $custom_field_listchuck = custom_chuck();

    //if ( isset($_POST['user_town']) ){
    update_post_meta($post_id, 'field_listchuck', $custom_field_listchuck);
    //}

}
add_action('save_post', 'save_field_custom');


function show_resultapi_in_woocommerce()
{

    $cf_listchuck = get_post_meta(get_the_ID(), 'field_listchuck', true);

    if ($cf_listchuck) {
        echo "Resultado de la implementación: " . $cf_listchuck;
    }
}

add_action('woocommerce_after_single_product_summary', 'show_resultapi_in_woocommerce', 20, 0);
```

esta función crea el campo personalizado

y se activa a través del hook save_post cada vez que se guarda un un post al editarlo o crearlo.

```php
function show_resultapi_in_woocommerce()
{

    $cf_listchuck = get_post_meta(get_the_ID(), 'field_listchuck', true);

    if ($cf_listchuck) {
        echo "Resultado de la implementación: " . $cf_listchuck;
    }
}

add_action('woocommerce_after_single_product_summary', 'show_resultapi_in_woocommerce', 20, 0);
```

Y esta función obtiene el contenido del campo personalizado si existe y lo pintamos en el front del producto cada vez que sea llamado por el hook de Woocomerce

Repositorio: https://github.com/Daljesone/Norris

Consultas al postman
Generamos nuestra rest api en woocomerce

Y generamos las consultas GET